Naval Research Laboratory

Washington, DC 20375-5320

NRL/FR/6110--98-9879

Probabilistic Neural Networks for Chemical Sensor Array Pattern Recognition: Comparison Studies, Improvements and Automated Outlier Rejection

Ronald Shaffer
Susan L. Rose-Pehrsson
R. Andrew McGill

Chemical Dynamic and Diagnostics Branch
Chemistry Division

March 10, 1998

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE March 10, 1998	3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE Probabilistic Neural Networks for Chemical Sensor Array Pattern Recognition: Comparison Studies, Improvements and Automated Outlier Rejection	5. FUNDING NUMBERS
6. AUTHOR(S) Ronald Shaffer, Susan L. Rose-Pehrsson, and R. Andrew McGill	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory Washington, DC 20375-5320	8. PERFORMING ORGANIZATION REPORT NUMBER NRL/FR/6110--98-9879
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force, HSC/YAEC Brooks Air Force Base, San Antonio, TX	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

For application to chemical sensor arrays, the ideal pattern recognition is accurate, fast, simple to train, robust to outliers, has low memory requirements, and has the ability to produce a measure of classification certainty. In this work, four data sets representing typical chemical sensor array data were used to compare seven pattern recognition algorithms—nearest neighbor, Mahalanobis linear discriminant analysis, Bayesian linear discriminant analysis, SIMCA, back propagation neural networks, probabilistic neural networks (PNN), and learning vector quantization (LVQ)—for their ability to meet the criteria. LVQ and PNN exhibited high classification accuracy and met many of the qualitative criteria for an ideal algorithm. Based on these results, a new algorithm (LVQ-PNN) that incorporates the best features of PNN and LVQ was developed. The LVQ-PNN algorithm was further improved by the addition of a faster training procedure. It was then compared with the other seven algorithms. The LVQ-PNN method achieved excellent classification performance. A general procedure for selecting the optimal rejection threshold for a PNN-based algorithm using Monte Carlo simulations also was demonstrated. This outlier rejection strategy was implemented for an LVQ-PNN classifier and found consistently to reject ambiguous patterns.

14. SUBJECT TERMS Neural network Pattern recognition Electronic nose	SAW Sensor array	Chemometrics Toxic vapor	15. NUMBER OF PAGES 33 16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std 239-18
298-102

CONTENTS

PROBABILISTIC NEURAL NETWORKS FOR CHEMICAL SENSOR ARRAY PATTERN RECOGNITION: COMPARISON STUDIES, IMPROVEMENTS AND AUTOMATED OUTLIER REJECTION

INTRODUCTION

Traditional chemical detection methods rely on the inherent selectivity of the sensor to provide the necessary information required to determine the presence or absence of the target analyte(s). Advancements in chemical sensor technology have allowed the measurement to move from the laboratory to the field. Field measurements offer additional challenges not seen in laboratory or controlled environments such as the need to detect target analyte(s) in the presence of large concentrations of interfering species. In an ideal situation, the chemical sensor responds only to the targeted analyte(s). However, many sensor technologies, such as polymer-coated surface acoustic wave (SAW) chemical sensors, cannot achieve this measure of selectivity. For the past 10 years, researchers have been able to overcome this potential drawback by coupling arrays of partially selective sensors with pattern recognition algorithms to interpret the complex sensor signals and provide an automated decision concerning the presence or absence of the targeted analyte(s). This approach has been used successfully for semiconducting gas oxide sensors [1], Taguchi gas sensors [2], MOSFET [3], electrochemical [4], and polymer-coated SAWs [5] for analyzing both liquid and gas phase species.

The underlying foundations for applying pattern recognition methods to chemical sensor arrays are:

1. the sensor signals numerically encode chemical information (i.e., a chemical signature) about the target analytes and the interfering species;

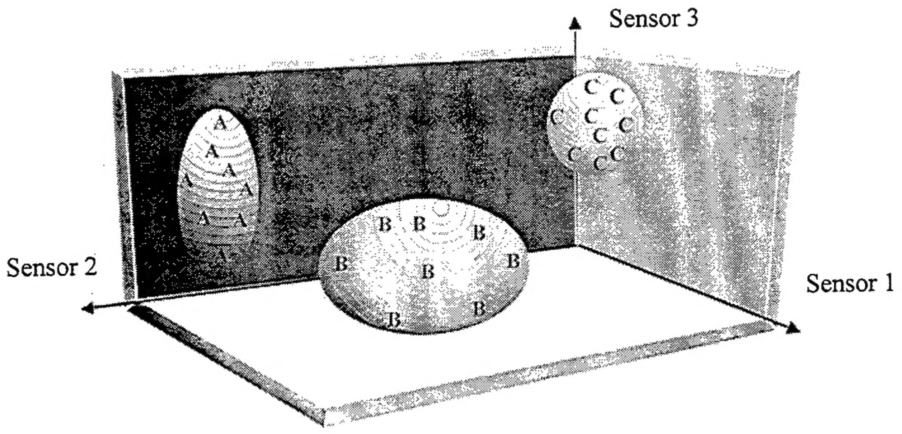
2. sufficient differences in the chemical signatures for the target analyte(s) and the interfering species can be exploited; and

3. the differences remain consistent over time.

For chemical sensor array pattern recognition, the responses for the *m* sensors in the array form an *m*-dimensional vector ("pattern") in the data space. Recognition of the chemical signature is based on the clustering of the patterns in the *m*-dimensional space (Fig. 1). Analytes that have different chemical signatures will cluster far away from each other in the pattern space; this allows them to be distinguished mathematically from other compounds.

Many researchers in the field of chemical sensors feel that between 10 and 20 partially selective sensors (i.e., a 10- or 20-dimensional pattern vector) is the maximum number of sensors that will be needed (or available) for most applications. In fact, many applications require less than five sensors to obtain the necessary chemical information (i.e., good clustering in multidimensional space). The desire to use the minimum number of sensors in a system is driven by the need to keep the instrumentation

Manuscript approved October 8, 1997.



3 Target Analytes (A, B, C)
3 Sensor Array (1, 2, 3)

Fig. 1 — Conceptual picture of a pattern space consisting of a three-sensor array and three classes of compounds

simple and reduce the size and cost of the system. The optimal number of sensors in the array is also determined by the specificity of the sensors themselves and the types of applications in which the sensors will be used. Various criteria for selecting the sensors in the array have been proposed [6-9].

In supervised pattern recognition methods, training patterns (i.e., chemical signatures) from known analytes and potential interfering species representative of the environment in which the sensor will be deployed are used to develop classification rules. These classification rules are used to predict the classification of future sensor array data. The training patterns are obtained by exposing the sensor array to both the target analyte(s) and potential interferents under a wide variety of conditions (e.g., varying concentrations, environments). The potential outcomes of the measurement (e.g., the presence or absence of a target analyte) are considered the data classes. The number of data classes is application-specific.

Many supervised pattern recognition algorithms have been used for the analysis of chemical sensor array data [1-15]. Compared to other applications, chemical sensor array pattern recognition has a unique set of requirements and goals. For example, many applications of pattern recognition in science and engineering require vast computational resources for learning the classification rules. Applications to infrared spectroscopy or the analysis of radar images often require pattern vectors of 100 or more dimensions. As discussed above, the pattern dimensionality for a sensor array is considerably smaller, typically ranging from 3 to 16, thus greatly decreasing the computation load on the pattern recognition algorithm. Also, for implementation in a field-portable sensor system, the algorithm must be transferable to a small microcontroller. Thus, many of the accepted practices that are true for pattern recognition in general may not be pertinent for chemical sensor array pattern recognition.

Based on the types of environments and situations in which chemical sensor arrays are expected to operate, the ideal pattern recognition algorithm should have the following six qualities:

1. *High accuracy*. For application of the chemical sensor system to field measurements, the pattern recognition algorithm must accurately classify new sensor signals (i.e., low false alarm

rate and ideally no missed detections). For military applications such as the detection of toxic chemical vapors, classification rates of greater than 90% correct are necessary.

2. *Fast.* For real-time analysis, the pattern recognition algorithm must be able to produce a classification quickly. Thus, algorithms that are computationally intense may not be appropriate for this application.

3. *Simple to train.* The classification rules used by the pattern recognition algorithm must be learned quickly. For many applications, the database of training patterns will need to be updated periodically, thus requiring the algorithm to "relearn" its classification rules. This procedure must be performed as simply and quickly as possible.

4. *Low memory requirements.* For field-portable sensor systems, the pattern recognition algorithm will be embedded in a microcontroller that has limited memory resources. Thus, pattern recognition algorithms with large memory requirements may not be appropriate for this application.

5. *Robust to outliers.* For chemical sensor array application in uncontrolled environments (e.g., battlefield), the pattern recognition algorithm must be able to reduce the potential for false alarms by being able to differentiate between sensor signals on which it was trained on and those on which it was not. The underlying assumption is that the algorithm has been trained to recognize all the important compounds. Thus, any new or ambiguous sensor signal (e.g., a chemical vapor that it has not been trained to recognize) should be ignored.

6. *Produce a measure of uncertainty.* For many applications of chemical sensor array systems, the pattern recognition algorithm must be able to produce a statistical measure concerning the certainty of the classification. For sensor applications such as toxic vapor monitoring, such a measure will aid in reducing the occurrence of false alarms by requiring that the sensor system be greater than 80% or 90% certain of a classification decision before a warning is given or an alarm sounded.

Unfortunately, no pattern recognition algorithm is able to fully meet each of these requirements. Most researchers use pattern recognition approaches based on either linear discriminant analysis (LDA) [5,10,11], principal components analysis (e.g., soft independent modeling of class analogy, SIMCA) [6], or back-propagation artificial neural networks (BP-ANN) [1,12-14]. Both LDA and SIMCA are computationally simpler and easier to train than a BP- ANN, but have trouble with multimodal and overlapping class distributions. BP-ANNs have become the de facto standard for chemical sensor pattern recognition as the result of the increasing power of personal computers and their inherent advantages in modeling complex data spaces. However, in recent work at the NRL using simulated chemical sensor array data, the probabilistic neural network (PNN) was shown to be a potentially powerful alternative to the conventional BP-ANN approaches [15].

The research described in this report is an extension of previous work performed at NRL applying PNN to chemical sensor array pattern recognition. The goals of this research project are three-fold:

1. compare PNN with other popular classification algorithms based on their ability to fit the criteria for the ideal chemical sensor array pattern recognition algorithm;

2. create an improved PNN algorithm that is faster and has lower memory requirements; and

3. develop a general procedure for rejecting ambiguous sensor signals by using the PNN.

EXPERIMENTAL

Four data sets representing typical chemical sensor array data were used in this study. Two of these data sets involved manufactured or simulated data; the remaining data were collected using SAW chemical sensor systems. The simulated data were obtained from Dr. Mark Anderson and were used in Ref. 15. The 960 pattern vectors in these data sets were generated by using a Gaussian probability distribution about a mean vector with a predetermined variance for each sensor. The mean vectors and variances were chosen to mimic the distribution of data obtained using an actual chemical sensor array. Training and prediction subsets were created by randomly subdividing the available pattern vectors (480 for training and 480 for prediction) while keeping the number of pattern vectors in each class equal (i.e., 80 and 120 patterns in each class for SIM1 and SIM2, respectively). The first simulated data set (SIM1) was based on a four-sensor array with six data classes. The second simulated data set (SIM2) was based on a six-sensor array with four data classes. The mean vectors and variances for each data class for SIM1 and SIM2 are listed in Tables 1 and 2, respectively.

Table 1 — SIM1 Data Set Class Statistics

Class	Mean Vector	Variance (10^{-3})
1	0.918, 0.302, 0.162, 0.106	60.8, 129.0, 70.2, 46.4
2	0.208, 0.849, 0.420, 0.105	97.1, 127.0, 146.0, 34.9
3	0.118, 0.751, 0.572, 0.232	44.4, 122.0, 147.0, 53.7
4	0.260, 0.370, 0.870, 0.123	106.0, 86.8, 64.0, 40.4
5	0.434, 0.269, 0.282, 0.776	132.0, 104.0, 120.0, 120.0
6	0.302, 0.575, 0.204, 0.682	108.0, 169.0, 56.5, 164.0

Table 2 — SIM2 Data Set Class Statistics

Class	Mean Vector	Variances (10^{-3})
1	0.665, 0.392, 0.352, 0.283, 0.276, 0.270	139.0, 85.0, 83.0, 75.0, 75.0, 81.0
2	0.247, 0.711, 0.369, 0.298, 0.303, 0.305	48.0, 85.0, 62.0, 58.0, 57.0, 57.0
3	0.280, 0.582, 0.370, 0.426, 0.347, 0.349	46.0, 65.0, 61.0, 71.0, 59.0, 57.0
4	0.282, 0.419, 0.365, 0.537, 0.338, 0.413	62.0, 80.0, 84.0, 118.0, 56.0, 64.0

The third data set (SAW1) was obtained from Tables 5 and 11 in Ref. 16. These data were collected using a four-sensor SAW array system with a preconcentrator sampling system. SAW data for two types of chemical warfare (CW) agents (blister and nerve agents), simulants, and potential interferents (nonagents) were collected under a variety of experimental conditions (e.g., changing humidity, concentrations and mixtures). In this study, the nerve agent class consisted of dimethyl methyl phosphonate (DMMP), pinacolyl methyl phosphofluoridate (GD), and O-ethyl-S-(2 isopropylaminoethyl) methyl phosphonothiolate (VX), while the blister agent class contained bis(2-chloroethyl) sulfide (HD) and dichloropentane (DCP). Toluene, water, diesel exhaust, jet fuel exhaust, dichloroethane, cigarette smoke, bleach, ammonia, sulfur dioxide, and isopropanol were placed in the nonagent class. Patterns obtained from SAW data of mixtures of a CW agent or simulant and a nonagent were placed in either the nerve or blister agent class (e.g., a pattern vector extracted from an exposure of a mixture of HD and diesel exhaust would be placed in the blister agent class). Because one of the sensors in the array did not respond (stopped oscillating) at high humidity, it was removed; this resulted in a three-dimensional pattern vector. References 5, 16, and 17 describe the SAW system, the methods of collecting the data, and the procedure for extracting pattern vectors from the raw SAW data. The patterns included in the prediction subset were obtained from SAW data collected at a different time than the training data and included some interferent vapors that were not in the training subset. Table 3 lists the number of pattern vectors in the training and prediction subset for this data set.

Table 3 — SAW1 Data Set

Class	Number of Patterns in the Training Subset	Number of Patterns in the Prediction Subset
Nerve	102	125
Blister	58	76
Non-agent	36	15
Total	196	216

The final data set (SAW2) consists of 664 pattern vectors obtained from SAW data collected using a six-sensor array with a preconcentrator sampling system. References 18 and 19 describe this SAW sensor system in detail. SAW data for seven chemical warfare agent or simulant vapors were collected. Some of these vapors (DMMP, GD, VX, and HD) were also included in SAW1. Other vapors in this data set included the nerve agents ethyl N,N - dimethylphosphoramidocyanidate (GA) and isopropyl methylphosphonofluoridate (GB). The blister agent simulant chloro ethyl ether (CEE) was also used in the SAW2 data set. For each vapor, three replicate exposures lasting greater than 10 minutes were performed at different humidity levels (ranging from 0 to 80%) and trace level concentrations. Reference 20 discusses the method for extracting pattern vectors from the raw SAW data. Similar to SAW1, one of the sensors in the array did not provide any additional information and was removed, resulting in a five-dimensional pattern vector. The 664 pattern vectors were subdivided into training and prediction subsets. Pattern vectors from replicate exposures were kept together to make the classification as realistic as possible. Because multiple concentrations were available for each vapor, the pattern vectors obtained from SAW data collected from an exposure to a midlevel concentration of each vapor were placed in the prediction subset. The remaining pattern vectors were placed in the training subset. Table 4 lists the vapors used in this study, the concentration for each set of exposures, and the number of patterns placed in the training and prediction subsets.

Table 4 — SAW2 Data Set

Classification	Concentration (ppb)	Concentration (mg/m^3)	Training Subset	Prediction Subset
GA	75, 151, 754	0.5, 1.0, 5.0	77	28
GB	9, 87, 175	0.05, 0.5, 1.0	56	37
GD	7, 67, 134, 644	0.05, 0.5, 1.0, 4.8	77	27
VX	1, 84	0.01, 0.92	49	17
HD	77, 307, 1537	0.5, 2.0, 10.0	67	30
DMMP	20, 99, 197, 1971	0.1, 0.5, 1.0, 10.0	51	48
CEE	171, 855, 1710, 8548	1.0, 5.0, 10.0, 50.0	50	50
Total			427	237

Each of the seven pattern recognition algorithms used in this study were implemented in MATLAB (version 4.2c, Mathworks, Inc., Natick, MA) on a Dell Pentium-120 personal computer running Windows 3.1 (Microsoft Corporation, Redmond, WA). The SIMCA routine was implemented using routines from the PLS_Toolbox (version 1.52, Eigenvector Technologies, Inc., Manson, WA). The learning vector quantization (LVQ) and BP-ANN algorithms were taken from the Neural Network toolbox (version 2.0A, Mathworks, Inc., Natick, MA).

RESULTS AND DISCUSSION

Comparison of Pattern Recognition Algorithms

Many researchers have performed studies comparing multiple pattern recognition algorithms. Derde and Massart performed a qualitative comparison of several classifiers popular in the general chemometrics community including SIMCA, LDA, and ALLOC (similar to PNN) [21]. Their comparisons were based on four technical aspects (optimal decision boundaries, overlapping regions, degree of certainty, and outliers) and four practical aspects (updates, variables of mixed type, irrelevant parameters, and ease of use) of supervised pattern recognition. They concluded that the choice of the best algorithm was application-specific and that hybrid approaches offered great potential. Probably the most comprehensive comparison study was published in a 1994 textbook by Michie, Spiegelhalter, and Taylor [22]. They studied several types of machine learning, statistical, and neural classification methods and compared them on both qualitative and quantitative aspects. Their suite of 23 pattern recognition algorithms contained several methods that are also included in this study (LDA, ALLOC, BP-ANN, and learning vector quantization (LVQ)). They found that, in general, neural network methods performed very well in terms of predictive performance but required long training times and an expert's intuition to implement. They also reported that each method has its own relative advantages and disadvantages and that the practitioner would be wise to choose the appropriate algorithm for the application at hand.

The two studies cited above contain very thorough comparisons of several pattern recognition approaches for general application. However, we are primarily interested in how these algorithms perform on data obtained from chemical sensor arrays. As outlined earlier, the classification of chemical sensor array data offers a unique situation compared to pattern recognition in general. In the first part of this research, seven pattern recognition algorithms are compared for their ability to meet the criteria for an ideal chemical sensor array pattern recognition algorithm. The seven pattern recognition algorithms chosen for this study represent some of the most commonly used approaches in the chemical sensor and pattern recognition communities.

When comparing classification algorithms, the formulation of the problem is critical. Applications of pattern recognition to chemical sensor arrays can be expressed as a multiclass problem (i.e., the predicted class of the pattern must be one of the two or more given categories), sometimes termed "winner-take-all (WTA)" or "1-of-*n*" coding, or as a series of two-class problems (i.e., target compound(s) are either present or absent). In terms of classification accuracy, some experimental evidence suggests that splitting the pattern recognition into a series of two-class or binary problems ("divide and conquer") improves prediction performance [23].

The primary disadvantage for formulating the classification as a series of binary decisions is that for applications involving multiple classes (e.g., each of the four data sets in this study), formulating several two-class problems becomes computationally cumbersome. For pattern recognition algorithms with long training times, expressing the application as a series of two-class problems may not be feasible. For this study, it was decided that a multiclass problem was the simplest approach.

A brief description of the operating principles of each algorithm is given below. References 21, 22, and 24-28 can be consulted for a more theoretical discussion of these algorithms and pattern recognition in general.

1. *Nearest Neighbor (NN)*. NN is a simple, nonparametric pattern recognition algorithm based on the Euclidean distance metric that can handle complex data spaces. An NN classifier requires no training. To classify a new pattern, the Euclidean distance between the new pattern and each pattern in the training set is computed. The Euclidean distance metric is given by

$$d_{ij} = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2}, \tag{1}$$

   where d is the Euclidean distance between patterns i and j, each with n variables (sensors). The classification of the new pattern is assigned to the classification of the closest pattern (i.e., smallest d) in the training set.

2. *Linear Discriminant Analysis (LDA)*. LDA is a computationally efficient, parametric pattern recognition algorithm. The LDA algorithm used in this work is based on the Mahalanobis distance metric [29]. An LDA classifier is trained by computing a mean vector for each class and the pooled covariance matrix. The mean vectors and covariance matrix define the class boundaries. To classify a new pattern, the Mahalanobis distance between the new pattern and the mean vector for each class is computed. The Mahalanobis distance metric is computed as

$$d_{ij} = \sqrt{(x_i - \overline{x}_j)' S^{-1} (x_i - \overline{x}_j)}, \tag{2}$$

   where d is the Mahalanobis distance between pattern vector i and the mean pattern vector for class j and S is the pooled covariance matrix. The classification of the new pattern is assigned to the classification of the closest (i.e., smallest Mahalanobis distance) mean vector.

3. *Bayes Linear Discriminant Analysis (BDA)*. BDA is another computationally efficient parametric pattern recognition algorithm. It is based on the Bayes strategy for minimizing risk associated with the classification decision. Training is performed by using the mean vector for each class and the pooled covariance matrix to position a linear separating surface, termed a discriminant, to satisfy the Bayes criteria [24]. This separating surface is defined by its normal vector, often termed a weight vector. For multicategory applications such as the four data sets discussed in this report, a single discriminant is computed for each class. The assignment of class membership for new patterns is based on the side of the discriminant in which the pattern vector lies. This can be determined by computing the scalar dot product of the pattern vector with each weight vector. The product of this calculation is termed the discriminant score. In a WTA formulation, the new pattern is assigned to the class whose discriminant generated the largest discriminant score [25,26].

4. *Soft Independent Modeling of Class Analogy (SIMCA)*. The SIMCA pattern recognition algorithm uses principal component analysis (PCA) to construct an individual multivariate model for each class [27,28]. The model for each class uses as many principal components as are necessary to describe that class adequately. Classification is performed by computing the residual between the new pattern and the model. Using a WTA strategy, the new pattern is assigned the classification of the model that produced the smallest residual.

5. *Artificial Neural Networks (BP-ANN)*. ANNs are nonlinear, nonparametric pattern recognition algorithms that typically use the back-propagation (BP) method for learning the classification rules (i.e., define the boundaries between the data classes) [22,26,30]. In most chemical sensor array applications, a BP-ANN consists of an input layer, one or two hidden layers, and an output layer of neurons. A neuron is simply a processing unit that outputs a linear or nonlinear transformation of its inputs (i.e., a weighted sum). The neurons, as a group, map the input pattern vectors to the desired outputs (the known data class of the pattern vector). In a 1-of-*n* coding scheme, there is one output neuron for each potential data class. Using BP, the weights and biases associated with the neurons are modified to minimize the mapping error (i.e., the training set classification error). Upon repeated presentation of the training patterns to the BP-ANN, the weights and biases of the neurons become stabilized and the ANN is said to be trained. New patterns are classified by propagating the new pattern through the neural network (i.e., multiplying by weights, adding biases, and applying the nonlinear transfer function). In a WTA formulation, the output neuron with the highest score indicates the classification of the new pattern.

6. *Probabilistic Neural Networks (PNN)*. The PNN is a nonlinear, nonparametric pattern recognition algorithm that operates by defining a probability density function (PDF) for each data class based on the training set data and the optimized kernel width (σ) parameter [15,30-33]. Each PDF is estimated by placing a Gaussian-shaped kernel at the location of each pattern in the training set. A multivariate estimate of the underlying PDF for each class can be expressed as the sum of the individual Gaussian kernels. The PDF defines the boundaries (i.e., the classification rules) for each data class. The kernel width determines the amount of interpolation that occurs between data classes that lie near each other in the data space. As the kernel width approaches zero, the PNN will produce the same classification decisions as an NN classifier. In this implementation, a single kernel width will be used to estimate each PDF. For some applications involving data where one class has a much different distribution than the other classes, a single kernel width may not be optimal. In these cases, the PNN can be modified to accept a different kernel width for each sensor or class [15,31]. These implementations were not included in this study. For classifying new patterns, the PDF is used to estimate the probability that the new pattern belongs to each data class. Figure 2 shows the topology of the PNN. PNN training is accomplished by simply copying each pattern in the training set to the hidden layer and optimizing σ. Cross-validation and univariate optimization are commonly used for choosing the best σ. The hidden layer neurons and σ can then be downloaded to the chemical sensor array system for use in the field. The classification of new patterns is performed by propagating the pattern vector through the PNN. The input layer is used to store the new pattern while it is serially passed through the hidden layer. At each neuron in the hidden layer, the distance (in this work the dot product distance) is computed between the new pattern and the training set pattern stored in that hidden neuron. The distance d is processed through a nonlinear transfer function

$$output = e^{-\left(\frac{1-d}{\sigma^2}\right)}. \tag{3}$$

The summation layer consists of one neuron for each data class, and it sums the outputs from all hidden neurons of each respective data class. The products of the summation layer are forwarded to the output layer (one neuron for each data class) where the estimated probability of the new pattern being a member of that data class is computed.

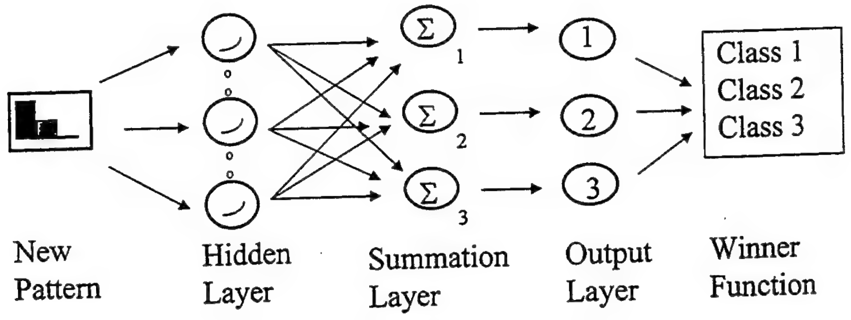


Fig. 2 — Topology of the probabilistic neural network (PNN). In this example, the inputs are the responses from a three-sensor array, and the PNN outputs are the predicted classification (either class 1, 2, or 3) for a given pattern.

7. *Learning Vector Quantization (LVQ)*. LVQ is another nonparametric pattern recognition algorithm that combines some of the features of NN and competitive learning ANNs [22,34]. The basic premise is that the underlying PDF for each class can be estimated by using a small number of references vectors (i.e., pattern vectors). LVQ uses a competitive learning algorithm to define a smaller set of reference vectors that span the same space as the original training set patterns. In this training scheme, the reference vectors are considered hidden neurons, are randomly assigned a classification, and are forced to compete against each other to learn the structure of the pattern space. The initial assignment of the classification for each reference vector is done using the relative distribution of the output data classes so that the hidden layer is a statistically accurate representation of the pattern space. Thus, if 25% of the pattern vectors in the training set belong to class 1, then 25% of the reference vectors are assigned to class 1. The initial location for each reference vector is the mean vector for the training subset. After initialization, the patterns in the training set are repeatedly presented to the hidden layer in random order. At each iteration, the hidden neuron closest (in terms of Euclidean distance (Eq. 1)) to the current pattern, termed the "winning" neuron, is moved toward the current pattern if the classification for the winning neuron matches the classification of the pattern, otherwise it is moved away from pattern. The classification of new patterns is performed the same as NN except the LVQ hidden layer replaces the training set.

Relative Advantages and Disadvantages

Table 5 compares the seven pattern recognition algorithms based on the guidelines or criteria for the ideal pattern recognition algorithm. The relative advantages and disadvantages of the seven approaches are discussed in more detail below.

In terms of the speed of operation, only NN and PNN do not meet the necessary requirements because the distance between the new pattern and every pattern in the training set (or hidden layer for PNN) must be performed for each calculation. Thus, this is a major drawback for applications requiring a large number of pattern vectors to adequately describe the data space. As discussed later in this report, however, this limitation can be partially overcome by judiciously choosing the pattern vector database.

Among the algorithms studied, BP-ANN is considered the slowest and most difficult to train. The BP training algorithm used in a BP-ANN is both slow and prone to local minima, requiring many replicate optimizations (sometimes called training runs). Also, the optimal architecture (e.g., number

Although many ANN researchers have attempted to overcome these limitations, there are no generally acceptable solutions to the problems. However, for applications involving simple pattern spaces, the choice of architecture is not critical and training speeds may be acceptable.

For most applications, SIMCA and LVQ training are both fast and straightforward. However, SIMCA relies on PCA for the critical model building step, and the choice of the optimal number of principal components may sometimes be difficult. Because LVQ training features thousands of random presentations of the data, the training may not reach convergence quickly, and sometimes it gets stuck in local optima. However, most researchers point out that these problems occur less frequently than with BP-ANN.

NN and PNN have larger memory requirements than the other pattern recognition algorithms in this study. Both approaches require that each pattern in the training set be stored for use in classifying new patterns. For applications involving a large number of sensors (i.e., high dimensionality of the pattern vector) or requiring many pattern vectors to describe the data space, the memory requirement may overcome the amount available on most microcontroller cards.

In terms of outlier rejection, any of the algorithms that can produce a measure of uncertainty have some limited outlier rejection capabilities. If the certainty for a classification is less than 80% or 90%, then the pattern can be rejected as an outlier. Distance-based approaches such NN, LDA, SIMCA, PNN, and LVQ offer additional outlier rejection power. In general, if the distance between the pattern being tested and the patterns in the training set is large, then the test pattern can be rejected as an outlier. However, in such distance-based schemes, the choice of a rejection criterion is not straightforward. For BP-ANN, one approach uses two ANNs, one for rejecting outliers and the second for classification [35]. Obviously, such an approach will not be feasible for many applications.

In general, all pattern recognition algorithms can produce some statistically based measure of uncertainty. Given enough pattern vectors, an external distribution of classification scores could be used in conjunction with any of the seven algorithms studied here to provide some confidence levels, but this requires extensive calculations and large amounts of available data [36]. However, for some algorithms, the ability to generate a statistically based measure of uncertainty is simple. The classification outputs of parametric methods such as LDA, BDA, and SIMCA and density-based methods such as PNN can be interpreted as a posteriori probabilities. Perhaps less well known, the other methods (NN, LVQ, and BP-ANN) can also use Bayes theorem to obtain statistically meaningful probabilities. The extensions developed for the NN routine to yield probabilistic information could probably be used for LVQ also [21]. Bishop discusses methods such as the softmax criterion and cross-entropy that allow the BP-ANN outputs to be interpreted as a posteriori probabilities [26]. Although possible, the use of probabilistic outputs in NN, LVQ, and BP-ANN is not common.

Table 5 — Qualitative Comparison of Pattern Recognition Algorithms

Pattern Recognition Algorithm	Speed of Operation	Simple and Fast to Train	Memory Requirements	Outlier Rejection Simple	Statistical Measure of Uncertainty
NN	slow	yes	high	yes	possible
LDA	fast	yes	low	yes	yes
BDA	fast	yes	low	sometimes	yes
SIMCA	fast	sometimes	low	yes	yes
BP-ANN	fast	no	low	sometimes	possible
PNN	slow	yes	high	yes	yes
LVQ	fast	sometimes	low	yes	possible

Classification Accuracy

To compare the classification accuracy of the seven pattern recognition algorithms in this study, the four data sets (Tables 1-4) were used. Where necessary, the pattern vectors in each data set were vector normalized to remove concentration information. Vector normalization was performed by dividing each element of the pattern by the square root of the sum of the squares. To be consistent, each algorithm used the WTA strategy for each classification decision.

The patterns in the training set were used to train the classifier (i.e., develop the classification rules), while the patterns in the prediction subset were withheld for use as a final validation step. For BP-ANN and LVQ, the training subset was further subdivided into a smaller training subset (80%) and a monitoring subset (20%). The patterns in the smaller training subset were used in training. At various intervals during the training procedure, the classification of the patterns in the monitoring set were predicted.

When comparing pattern recognition algorithms, it is imperative that the predictive performance be an unbiased indication of what can be expected from the classifier when used on the general population (i.e., used outside of the laboratory). Because some of the algorithms studied here require the optimization of classification parameters (e.g., the number of hidden neurons in BP-ANN) in addition to regular training, multiple training runs will need to be performed. As pointed out by Masters [30] and many others, the conclusions will be *biased* if the prediction set (Masters calls this the validation set) is used to decide which set of conditions is optimal. Thus, for each algorithm, the best classifier is chosen based solely on the results from either the training set, cross-validation of the training subset, or through a combination of training and monitoring subsets. Once the optimal classifier for each pattern recognition algorithm is chosen, the percentage of patterns correctly identified in the prediction subset can be used as an unbiased measure of predictive performance.

Because NN and PNN directly use each pattern in the training set to classify new patterns, the training subset classification performance (defined here as the percentage of pattern correctly identified) is statistically biased. The method of leave-one-out cross-validation (CV) was used for PNN and NN to obtain an unbiased measure of classification performance for the training subset [30]. CV involves many iterations of training and testing. For each iteration, the pattern recognition algorithm (either PNN or NN in this case) is trained using all the patterns in the training subset except one (i.e., the "holdout pattern"). The trained algorithm is then used to predict the classification of the holdout pattern. This process is repeated until each pattern has been held out once.

The SIMCA and neural network (LVQ and BP-ANN) pattern algorithms have many configuration options. To make this comparison as fair as possible, several configuration options for each algorithm were used. The configuration options that resulted in the best classification performances on the training and monitoring subset data were considered the optimal configurations. For SIMCA, models based on two principal components for each class were found to be the optimal configuration for each data set.

The optimal PNN configuration was determined by selecting the kernel width σ value that minimized the cross-validation classification error using

$$error = N_m + \sum_{i=1}^{n} (1 - p_i)^2 \,, \tag{4}$$

where N_m is the total number of patterns misclassified, n is the number of patterns in the training subset, and p_i is the PNN classification probability (i.e., measure of certainty) for the known classification. Thus, in this procedure (CV-OPT), if every pattern in the CV subset were correctly classified with no uncertainty, the error would be equal to 0. The minimization procedure used in this work is based on a parabolic interpolation method similar to the procedure discussed in Masters [31] and Brent's method [37]. In this approach, the kernel width is modified at each iteration to reduce the cross-validation training error (Eq. 4). In this work, 10 iterations were found to be sufficient to achieve convergence. The optimal kernel widths for the four data sets in this study were found to be 0.0358, 0.0678, 0.0138, and 0.0062 for SIM1, SIM2, SAW1, and SAW2, respectively. The dot product calculation, which is considered more efficient than other distance measures such as the Euclidean distance in cases where the patterns are already vector normalized, was used in the PNN algorithm for this work.

The LVQ training procedure used in this work consisted of 10,000 epochs (random presentations). The key parameters for LVQ are the number of hidden layer neurons and the learning rate. In this work, the learning rate was set initially to 0.05 and decreased by 10% every 500 epochs. The number of hidden layer neurons was varied from 25 to 150. Each training run was replicated three times to ensure convergence. For each training run and configuration, the neural network that produced the lowest training and monitoring set classification error was selected. For the four data sets in this study, 120, 120, 25, and 25 hidden units for SIM1, SIM2, SAW1, and SAW2, respectively were found to be optimal. The number of hidden neurons necessary to describe the SAW data was less because these data sets featured several replicate pattern vectors for each vapor.

The BP-ANN training procedure used the Levenberg-Marquardt technique to optimize the weights of the hidden layer. The training was performed for 500 epochs. Every 25 epochs, training was stopped and the patterns in the monitoring set were classified. In this work, one hidden layer was used and the number of hidden units was varied from 3 to 7. It was observed that the BP-ANN training was easily trapped in local minima, thus requiring that the training runs for each configuration be repeated 10 times to increase the probability of convergence. The neural network that misclassified the least number of patterns in the training and monitoring subsets was selected for each data set. For the four data sets in this study, 5, 4, 3, and 3 hidden units for SIM1, SIM2, SAW1, and SAW2, respectively, were found to be optimal.

Table 6 lists the percentages of patterns in the training set that were correctly classified for each pattern recognition algorithm in each data set. Ideally, the percentage of patterns correctly classified in training is a good measure of the predictive ability of the algorithm. However, it was noticed that, despite having the best training results, the performance of BP-ANN in prediction was no better than the other algorithms. The slight drop-off in classification performance between the training and prediction data suggests that overtraining might have occurred. Overtraining results from using too large a network or by simply memorizing the members of the training set. In such cases, the neural network does very well on the training data, but fails to predict the classification of the patterns in the prediction subset. However, in this case, ample measures were taken to prevent overtraining (e.g., using both a small number of hidden neurons and the network with the minimum training and monitoring error). Because the degree of overtraining was minimal, we feel that for these data sets it was unavoidable. Perhaps using another ANN architecture such as radial basis function (RBF) or cascade correlation would alleviate this problem. However, these methods are relatively new and were not studied in this report.

Table 6 — Percentage of Pattern Correctly Classified in the Training Subset

	PNN	BP-ANN	LVQ	SIMCA	NN	LDA	BDA
SIM1	93.75	98.17	94.53	95.63	91.88	90.21	84.79
SIM2	85.42	90.89	89.06	84.58	79.58	83.54	83.33
SAW1	96.94	100.0	97.44	88.27	97.45	88.27	90.31
SAW2	96.96	97.04	94.67	93.44	96.96	79.93	84.07

Table 7 lists the percentages of patterns in the prediction set that were correctly classified for each algorithm and data set. The percentages that are listed in bold typeface are the prediction performance for each data set that was judged by the authors as the best. The criteria used to make this decision for each data set are discussed in the following sections.

Table 7 — Percentage of Pattern Correctly Classified in the Prediction Subset

	PNN	BP-ANN	LVQ	SIMCA	NN	LDA	BDA
SIM1	93.13	92.92	93.54	**94.38**	92.08	90.63	85.83
SIM2	82.50	82.50	**83.54**	83.33	76.46	81.46	80.63
SAW1	90.74	**95.37**	94.91	85.65	90.74	62.50	96.76
SAW2	94.09	93.25	**94.94**	86.92	93.67	86.08	89.87

The SIM1 data set featured six data classes that were slightly overlapped. A large number of patterns for each data class were available and were normally distributed about the mean vector (see Table 1). Thus, SIMCA, PNN, and LDA would be expected to perform very well because they either presume an underlying distribution or attempt to model it. However, the prediction results did not follow this expected trend exactly. SIMCA did perform the best and PNN did very well, but LDA did not achieve satisfactory classification results. The LVQ approach was able to select the important pattern vectors that occur near the class boundaries and performed very well in prediction.

The SIM2 data set featured four data classes that were highly overlapped. Analogous to SIM1, a large number of patterns were available for each class and were normally distributed about the mean vector (see Table 2). For this data set, the LVQ and SIMCA algorithms perform very well in prediction, similar to the results for SIM1. NN performed the worst on this data set.

The SAW1 data set contained pattern vectors for three classes of vapor data (nerve, blister, or interferent). Unlike the simulated data sets (SIM1 and SIM2), the pattern vectors for each class are not Gaussian distributed around a mean vector, but have considerable scatter because of experimental error and multimodality caused by having different chemical vapors within each data class. In addition, this data set contained the smallest number of training patterns. Figure 3 is a plot of the 196 pattern vectors in the training subset projected onto their first two principal components (explains 97.58% of the variance), illustrating the scatter and multimodality present in the pattern space. The prediction set for SAW1 contains pattern vectors obtained from both later experiments and other vapors not included in the training subset. Figure 4 is a plot of the 216 pattern vectors in the prediction subset projected onto the first two principal components of the training subset. It is evident from these plots that the pattern vectors for the training and prediction subsets do not match up perfectly. However, this is often the case with real-world applications because of the constantly changing environments found in field measurements.

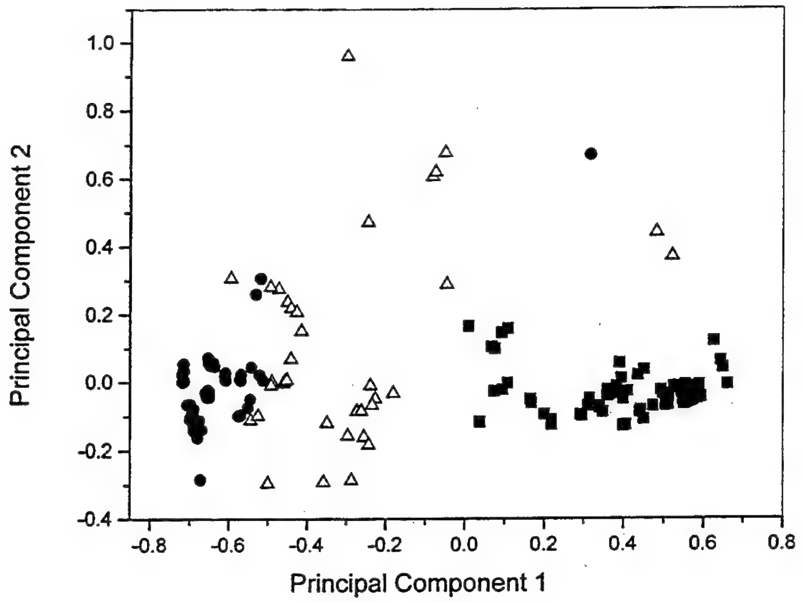


Fig. 3 — Principal components scores plot of the 196 patterns in the training subset of SAW1 (nerve: ■; blister: ●; nonagent: Δ)

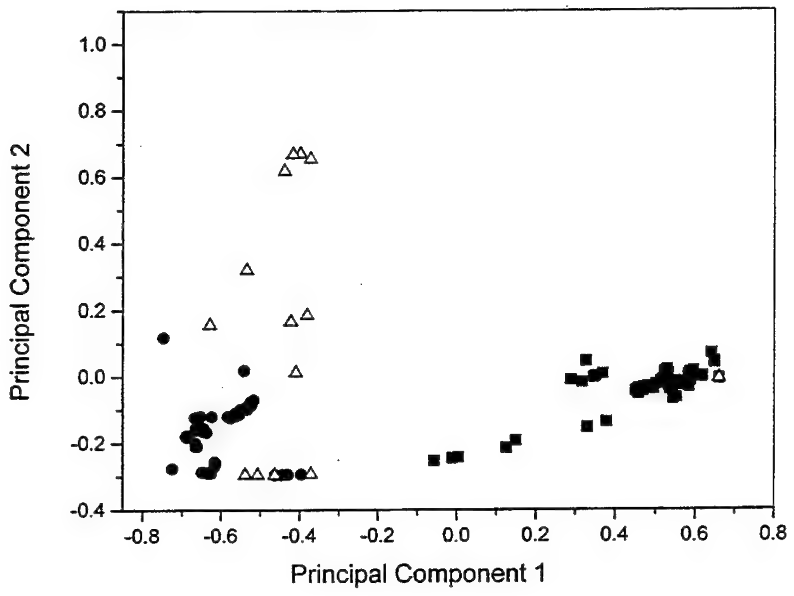


Fig. 4 — Plot of the 216 patterns in the prediction subset of the SAW1 projected on the first two principal components in the training subset (Fig. 3) (nerve: ■; blister: ●; nonagent: Δ)

Applications involving multimodal classes require a nonlinear boundary between the classes. Thus, this data set appears to favor the neural network and nearest neighbor algorithms. However, the prediction classification results indicated that BDA performed the best. Its prediction performance is much better than its relatively poor classification (90.3%) of the patterns in the training subset seems to suggest. Further study of this anomaly indicated that this result was because the training and prediction subsets do not cover the same data space exactly. The position of the discriminant computed by the BDA procedure was such that most of the patterns that it missed in the training subset were not included in the prediction subset. Thus, the prediction classification performance for BDA can be considered inflated.

With this information in mind, the BP-ANN and LVQ approaches performed the best on this data set. PNN and NN performed very well on the training set data. Although they did not do as well on the prediction data, the percentage of patterns correctly identified was greater than 90%. For this data set, the BP-ANN and LVQ algorithms did a better job of learning the underlying data structure and overtraining was not a major problem.

The SAW2 data set included seven data classes, several of which were multimodal and featured highly overlapping clusters. However, unlike SAW1, plenty of training patterns were available for learning the structure of the data space. Figure 5 is a plot of the 427 pattern vectors in the training subset projected onto their first two principal components (explains 92.13% of the variance), illustrating the overlapping clusters for the GB and GD classes and the scatter and multimodality present in the pattern space. The prediction set for SAW2 contains pattern vectors obtained from different vapor concentrations. Figure 6 is a plot of the 237 pattern vectors in the prediction subset projected onto the first two principal components of the training subset. Because of the multimodality and nonlinear nature of the pattern space, neural networks and NN should exhibit good classification ability. However, since the training and prediction subsets are similar and more training patterns are available, the PNN and NN classification performances should improve over the SAW1 results. For this data set, LVQ performed the best, with PNN, NN, and BP-ANN also doing very well. SIMCA, BDA, and LDA all had trouble defining classification boundaries due to the multimodal nature of some of the data classes, resulting in poor classification ability. For multimodal distributions, the assumption used in LDA and BDA that the structure of the covariance matrix for each class is similar, is no longer satisfied.

Conclusions

Among the algorithms studied here, it appears that the neural network based approaches (LVQ, BP-ANN, and PNN) are the most accurate classifiers for typical chemical sensor array data. This conclusion will not surprise anyone in the chemical sensor community. Even for the applications where the pattern vectors are normally distributed about a mean vector (e.g., SIM1 and SIM2), the neural network approaches still classify as accurately as the other approaches. It is interesting that LVQ had the best classification performance on the prediction subset for three of the four data sets in this study. After consideration of the other features of an ideal pattern recognition algorithm (Table 5), LVQ is still considered the best. A recent paper by Jurs et al. offers further experimental evidence for this conclusion [38]. The criterion that LVQ fails is that it has no simple statistical confidence measure for each classification decision. The only approach that can perform this task is PNN, which also does very well in terms of classification accuracy. Also, compared to PNN, LVQ training is considerably slower. Thus, depending on the needs of the particular application, we recommend PNN for applications where the speed and memory requirements are not critical but a confidence measure and fast training is, while LVQ is suggested as the algorithm of choice for all other applications. BP-ANN is a useful alternative to LVQ, but not until great improvements in training speed and simplicity are made widespread can we fully recommend it.

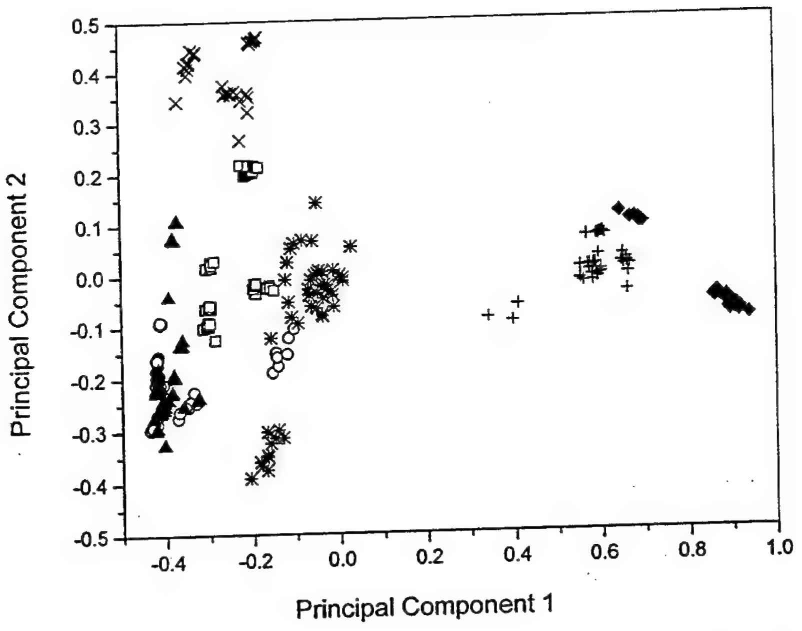


Fig. 5 — Principal components scores plot of the 427 patterns in the training subset of SAW2 (GA: □; GB: O; GD: ▲; VX: ✻; HD: ◆; CEE: +; DMMP: ×)

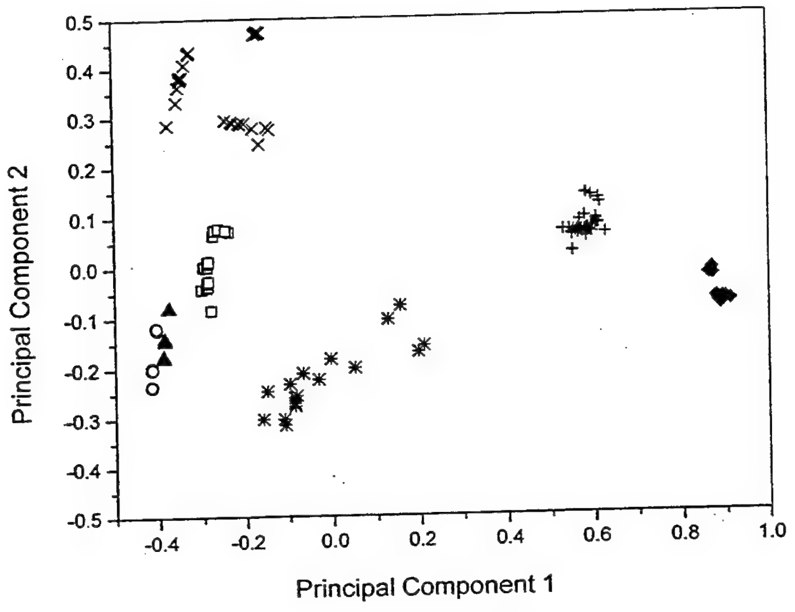


Fig. 6 — Plot of the 237 patterns in the prediction subset of the SAW2 projected on the first two principal components in the training subset (Fig. 5). (GA: □; GB: O; GD: ▲; VX: ✻; HD: ◆; CEE: +; DMMP: ×)

An Improved PNN Algorithm

As the comparison study suggested, PNN and LVQ appear to be the best pattern recognition algorithms for chemical sensor arrays. The primary drawbacks to the PNN approach are the slower speed of operation and higher memory requirements. LVQ lacks a simple method for measuring the confidence of a classification decision. Several researchers have noted that one method for improving the PNN algorithm would be to reduce the size of the hidden layer (i.e., the training subset). If fewer patterns were stored in the hidden layer, the memory requirements and the number of distance calculations would be reduced greatly.

One solution, suggested by Burrascano [39], was to use the optimized neurons of the LVQ hidden layer as the hidden layer of the PNN. As pointed out earlier, the LVQ hidden layer forms a good approximation of the data space using a minimal number of reference vectors, thereby making it a good candidate to replace the entire training set in the hidden layer of the PNN. Compared to conventional LVQ, the combined LVQ-PNN approach simply replaces the nearest neighbor decision with a PNN decision. In the comparison study, it was noted that the PNN approach had slightly better predictive performance than NN. Thus, replacing the NN step in LVQ with a PNN classifier in LVQ-PNN should result in a slight improvement in terms of classification performance. Burrascano found that for manufactured data sets, the combined LVQ-PNN approach reduced the memory requirements and improved the speed of operation of the PNN, while keeping the training times short and retaining the ability to produce a confidence measure for each classification [39].

An alternative approach to reducing the size of the hidden layer was proposed by Chtioui et al. [40]. They used a combination of PCA and reciprocal neighbors (RN) hierarchical clustering in the analysis of color images. In their work, the PCA step was needed to select a subset of relevant features (i.e., dimensionality reduction) from the vast amounts of image data that are collected. Dimensionality reduction is usually not necessary in our application because of the small number of sensors in the array. RN was used to select a small number of prototype vectors or patterns to replace the original training set. The RN algorithm is similar in operation to LVQ, but it is sensitive to the order in which the training data are presented.

Another issue in improving the basic PNN algorithm is the training speed. One of the current strengths of PNN is the ease and speed of training. The only adjustable parameter is the kernel width σ. The approach that has been used at NRL to select the optimal kernel width is a combination of cross-validation and univariate optimization (CV-OPT). However, the time needed to optimize the kernel width increases exponentially with the number of training set patterns. Many researchers have shown that PNN is essentially a neural network implementation of the classical kernel discriminant analysis method. In the 1960s and 70s, much research into the theory of probability density functions was published in the statistics literature [33]. One of the products of this research was various methods for approximating the optimal kernel width for kernel discriminant algorithms such as the Parzen classifier. Recently, Kraaijveld reported a simple distance-based calculation that seems to be a good approximation of the optimal kernel width for use in a Parzen classifier [41].

In the second part of this research, the potential of the LVQ-PNN approach for chemical sensor array pattern recognition was explored. In addition, studies to determine if the distance-based approximation to the optimal kernel width could be extended to the PNN paradigm were performed. A combination of LVQ-PNN and a faster training algorithm would have tremendous impact in chemical sensor pattern recognition. It would retain the advantages of both approaches and fulfill each criterion for an ideal pattern recognition algorithm. To study these two potential improvements, the four data sets used in the first part of this report were used.

Optimal Kernel Width Calculation

The equation used in this work for the optimal kernel width calculation is

$$\sigma_{opt} = 1.44 \sqrt{\frac{1}{n \cdot m} \sum_{j=1}^{n} \left\| \varepsilon_j^* - \varepsilon \right\|^2} \quad , \tag{5}$$

where m is the number of sensors in the array, n is the number of patterns in the hidden layer of the PNN, and ε_j^* represents the nearest neighbor of pattern ε_j. Kraaijveld suggested a correction factor in the range of 1.2 to 1.5 be used in calculating the optimal kernel width [41]. The 1.44 term in Eq. (5) is the correction factor that was found to work well with sensor array data. The basic assumption used in this approximation is that because the PDF for each class is estimated as the sum of the individual Gaussian kernels, the density estimate on a specific location in the pattern space is determined by the nearest kernel only. All other kernels are assumed to be so far away that their contribution to the density estimate is minimal. Thus, the approximation to the optimal kernel width should be based on the mean distance between nearest neighbors adjusted for the number of sensors and the number of patterns in the data set. For clarity, PNN or LVQ-PNN classifiers that use Eq. (5) for computing the optimal kernel width are termed PNN5 and LVQ-PNN5, respectively.

Classification Results

Table 8 lists the kernel widths and percentages of patterns correctly classified in the prediction subset using both the CV-OPT procedure and PNN5 for the four data sets used in this study. The PNN prediction results using the CV-OPT method of selecting the kernel width are also found in Table 7.

Table 8 — Optimized Kernel Widths and Prediction Classification Results for PNN

Data Set	CV-OPT Kernel Width	PNN Prediction	Eq. (5) Kernel Width	PNN5 Prediction
SIM1	0.0358	93.13	0.0274	92.29
SIM2	0.0678	82.50	0.0316	79.79
SAW1	0.0138	90.74	0.0317	90.74
SAW2	0.0062	94.09	0.0076	94.51

Discussion

These results suggest that Eq. (5) is a valid approximation to the optimal kernel width. For one data set (SAW2), the prediction results improved; for the two simulated data sets, the classification was not as successful. For two of the four data sets, the kernel width found by the two methods was similar and thus produced similar classification results. However, the speed and ease of implementing PNN5 justifies its further study. The CV-OPT method requires some user input (e.g., to set boundaries and to know when to stop), while the use of Eq. (5) is totally autonomous.

Implementation of LVQ-PNN

To make the entire LVQ-PNN training autonomous, several changes were made to the procedure used by Burrascano. First, the importance of the decision concerning the optimal number of hidden layer neurons has been decreased by always using a larger than necessary number of hidden neurons initially. LVQ training is performed as discussed earlier. Once training is complete, the network structure is interrogated by passing each pattern in the training and monitoring subset through the hidden layer of the LVQ once and storing each winning neuron. The neurons in the hidden layer that never become a winning neuron are considered "dead neurons" and are removed from the hidden layer.

The LVQ algorithm implemented in the neural network toolbox from MATLAB uses a time bias to encourage neurons that have not been chosen as a winner to have a higher score, hence improving their chances of being a winner [42]. Despite this adjustment, when a large number of initial neurons are used, many of them do not give any new information (i.e., cover some important section of the data space). Thus, pruning these neurons from the network does not hurt network performance greatly. It has the additional benefits of allowing the LVQ training to proceed without user intervention and further decreasing the size of the hidden layer passed to the PNN. If no dead neurons are removed, all of the neurons are significant. If this occurs and training results are not satisfactory, the initial number of neurons should be increased.

Classification Results

The LVQ-PNN5 algorithm was used to develop a classifier for the four data sets. To ensure that overtraining of the LVQ hidden layer does not occur, separate training and monitoring subsets were used (as was done earlier in this report). To determine the stability of these optimizations, each experiment was performed five times. The initial number of hidden neurons was set to 100 and 50 for the simulated (SIM1 and SIM2) and the real (SAW1 and SAW2) data sets, respectively. The LVQ training procedure was performed for 10,000 epochs, stopping every 500 to predict the patterns in the monitoring subset. The hidden neurons at the minimum training and monitoring set classification error were used as the input to the PNN, and Eq. (5) was used to compute the optimal kernel width. Tables 9-12 list the optimized kernel width, size of the hidden layer (i.e., number of "winning neurons"), and the percentages of patterns correctly classified in the training and prediction subsets for the standard LVQ and LVQ-PNN5 algorithms for each data set and training run. For each group of replicate training runs, the set of hidden neurons and kernel width that produced the best LVQ and LVQ-PNN5 training results was selected as optimal. As discussed in the comparison study, the prediction performance cannot be used to choose the best configuration without biasing the conclusions. Thus, the percentage of patterns correctly classified by LVQ in training was considered the first criteria and LVQ-PNN5 training performance was only used to break ties. In Tables 9-12, the training run (i.e., hidden layer and kernel width) that was judged optimal by this criterion is highlighted by bold lettering.

Table 9 — Classification Performance of LVQ and LVQ-PNN5 for SIM1

Run Number	LVQ Training	LVQ Prediction	Winning Neurons	LVQ-PNN5 Training	LVQ-PNN5 Prediction	Kernel Width
1	95.31	93.54	69	94.79	93.33	0.0394
2	94.79	94.17	82	94.79	94.17	0.0365
3	94.79	94.17	87	94.27	94.17	0.0339
4	**95.31**	**94.38**	**86**	**95.31**	**94.38**	**0.0344**
5	95.05	93.13	71	94.53	93.33	0.0397

Table 10 — Classification Performance of LVQ and LVQ-PNN5 for SIM

Run Number	LVQ Training	LVQ Prediction	Winning Neurons	LVQ-PNN5 Training	LVQ-PNN5 Prediction	Kernel Width
1	88.54	83.13	89	87.50	83.75	0.0267
2	**88.54**	**81.88**	**88**	**88.54**	**82.29**	**0.0273**
3	88.02	82.29	85	87.50	82.71	0.0259
4	88.28	82.71	66	87.50	82.71	0.0265
5	88.28	82.29	90	87.76	83.13	0.0269

Table 11 — Classification Performance of LVQ and LVQ-PNN5 for SAW1

Run Number	LVQ Training	LVQ Prediction	Winning Neurons	LVQ-PNN5 Training	LVQ-PNN5 Prediction	Kernel Width
1	97.44	93.98	29	96.15	93.52	0.0605
2	**98.08**	**94.44**	**36**	**96.79**	**94.44**	**0.0457**
3	98.08	93.98	39	94.87	93.52	0.0492
4	98.08	93.52	33	94.87	93.52	0.0473
5	97.44	94.44	37	96.79	94.44	0.0445

Table 12 — Classification Performance of LVQ and LVQ-PNN5 for SAW2

Run Number	LVQ Training	LVQ Prediction	Winning Neurons	LVQ-PNN5 Training	LVQ-PNN5 Prediction	Kernel Width
1	95.86	94.51	31	96.96	87.34	0.0454
2	96.45	94.94	34	92.04	96.20	0.0442
3	**97.34**	**94.94**	**32**	**92.97**	**96.20**	**0.0465**
4	96.15	94.94	32	96.72	86.50	0.0525
5	96.45	94.51	30	96.96	94.51	0.0469

Discussion

These results demonstrate that the LVQ-PNN5 approach produces a very good classifier of chemical sensor array data. The best LVQ-PNN5 prediction performances (94.38% for SIM1, 82.29% for SIM2, 94.44% for SAW1, and 96.20% for SAW2) compare favorably with the results from the comparison study (Table 7). Compared to those seven algorithms, the LVQ-PNN5 approach does better in prediction for SIM1 and SAW2, and only slightly worse than the best algorithm on the other two data sets. When compared to conventional PNN, in addition to better predictive performance, this new approach greatly reduces the memory and computation requirements. The number of patterns in the hidden layer was reduced on average to 15% of its original size (e.g., for SAW2, the size of the PNN hidden layer was reduced from 427 to 32 hidden neurons). The LVQ-PNN5 approach is also an improvement over the plain LVQ algorithm because of its slightly better prediction performance and simple method for generating statistically significant confidence measures.

As expected, the LVQ-PNN5 classification performances are fairly repeatable. The prediction results vary ±1% across the five training runs for three of the four data sets. However, as evident in Table 12, the percentages of patterns correctly identified in the prediction subset for the different training runs for the SAW2 data set had great variation in them. There are two potential causes for this phenomenon:

1. it could be due to the LVQ training step not always producing a good set of patterns (i.e., the hidden neurons) for the PNN classifier; or

2. because the procedure for computing the optimal kernel width (Eq. (5)) was not valid. To further investigate this issue, the CV-OPT method for determining the optimal kernel width was used on the five optimized hidden layers from the LVQ training step for the SAW2 data set. Table 13 lists the optimal kernel widths selected by this method and the percentages of patterns correctly classified by the LVQ-PNN using those kernel widths.

Table 13 — CV-OPT LVQ-PNN Classification Results

Training Run	Kernel Width	LVQ-PNN Prediction
1	0.0694	86.50
2	0.0948	91.98
3	0.3275	90.30
4	0.0207	94.94
5	0.0235	91.98

These results demonstrate that the poor prediction classification performance seen for run 1 (Table 12) using Eq. (5) to select the kernel width was not due to a poorly chosen kernel width since the prediction results did not improve using CV-OPT. Also, the kernel widths selected by both methods were similar (0.0694 and 0.0454). For the LVQ hidden layer from run 4, the LVQ-PNN did see a major improvement in predictive performance (87% to 95%), indicating that the poor performance of LVQ-PNN5 was due to a poorly chosen kernel width. In this case, the kernel width selected using Eq. (5) was 0.0465 but the kernel width selected using CV-OPT was 0.3275. Thus, it appears that in a few isolated cases the use of Eq. (5) to select the kernel width could hinder prediction performance. However, when we compare the prediction performance across all five runs (Tables 12 and 13), the PNN trained using CV-OPT actually performed worse than PNN5 for four of the five LVQ hidden layers. Thus, in general, poor predictive performance of the LVQ-PNN5 approach would be more likely due to a poor match between the LVQ selected hidden layer and the PNN classifier than a poorly chosen kernel width.

It is interesting to note that for 7 of the 20 runs across all four data sets (Tables 9-12) the LVQ-PNN5 classifier (sixth column from the right) outperformed and in 8 cases equaled the performance of the LVQ classifier (third column from the right) (remember than the LVQ classifier uses the nearest neighbor (NN) decision rule) using the same hidden layer. If the use of Eq. (5) to compute the optimal kernel width was the major cause of the poor predictive performance of LVQ-PNN5, the number of cases where LVQ prediction is better then LVQ-PNN5 prediction would be higher. This provides further justification for our belief that the primary cause of the few cases of poor classification performance for LVQ-PNN5 is due to a bad match between the LVQ hidden layer and the PNN classifier. Perhaps this might occur less frequently if some aspects of the PNN were included in the LVQ hidden layer competition. However, this would further increase the training times, and since the

PNN5 seems to be a good match in 15 of the 20 cases (Tables 9-12, columns 3 and 6), is probably not necessary.

To better understand how the LVQ training step works, the optimized hidden layer for run 3 in SAW2 was studied. Figure 7 is a plot of the hidden layer reference vectors projected onto the first two principal components of the SAW2 training subset (Fig. 5), analogous to the procedure used to create Fig. 6. Comparing Figs. 5 and 7 easily shows that 32 reference vectors in the LVQ hidden layer form a good approximation to the full training set (427 patterns). In cases where tight clustering is observed using the full training set (e.g., DMMP), only a few LVQ hidden neurons are required to approximate the cluster, while for less structured clusters (e.g., VX) or highly overlapping clusters (e.g., GB and GD) more LVQ neurons are required. Thus, an optimized LVQ hidden layer is a good hidden layer for the PNN, resulting in acceptable classification performance.

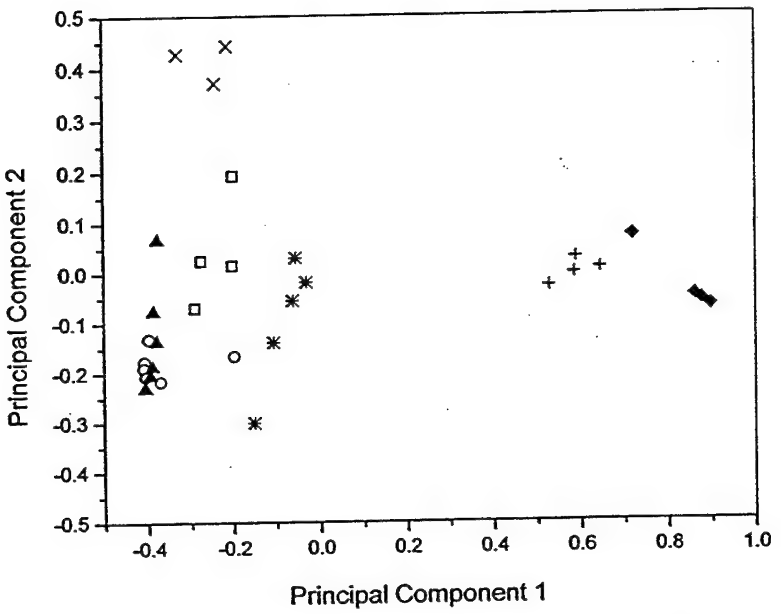


Fig. 7 — Plot of the 32 reference vectors in the optimized LVQ hidden layer for run 3 in SAW2 projected on the first two principal components in the training subset (Fig. 5). (GA: □; GB: O; GD: ▲; VX: ✳; HD: ◆; CEE: +; DMMP: ×)

Conclusions

These studies show that using LVQ to select an optimized hidden layer and a PNN using Eq. (5) to compute the kernel width (LVQ-PNN5) for classification produces better than or equivalent results to the LVQ, PNN, or PNN5 algorithms alone. The advantages to coupling these algorithms far outweigh the disadvantages. The LVQ-PNN5 approach can be implemented in an autonomous fashion, thereby greatly increasing its usability. In terms of the ideal pattern recognition algorithm, this new approach is better suited to meet those criteria.

However, there is still more work to be done on this approach. The coupling of the LVQ and PNN5 approaches is sometimes not a good match. In a few of these isolated incidences, Eq. (5) does not appear to be a very good approximation for the optimal kernel width. Further understanding of this phenomena is needed. Also, the scalability of Eq. (5) is not known. In the four data sets studied here, the dimensionality of the pattern vectors was very small, ranging from 3 to 6. Although Eq. (5) has a correction term for the dimensionality *m*, it is not known whether it will still be a good approximation. For higher dimension data, the chances of having redundant information increases greatly, which will cause problems with the distance calculation. Further experimentation needs to be done in this area as well.

Outlier Rejection Using the PNN

For chemical sensor array systems in real-world environments, the ability to reject ambiguous sensor signals (i.e., patterns) is critical. Ambiguous patterns are encountered for two main reasons. They may occur when an analyte belonging to a class of compounds that was not included in the training set (data base) is encountered (e.g., for a SAW chemical sensor system designed to detect toxic vapors, a new type of perfume) or when rapid changes in the environment surrounding the sensor system (e.g., rapid change in temperature or humidity) cause random fluctuations in the sensor signal. As discussed in point 5 in the list of criteria for an ideal classification algorithm, the pattern recognition technique must be "smart" enough to know when it does not know how to classify something. Most pattern recognition algorithms, including the ones in this study, force the algorithm to make a choice (i.e., WTA). By requiring the algorithm to place a new pattern into one of the known classes, the chances of false alarms increase for ambiguous patterns. Teaching the algorithm to classify the pattern as an unknown is a difficult task [35].

Pattern recognition algorithms based on a distance measure, such as PNN, have the potential to flag ambiguous patterns. In terms of statistical pattern recognition, an ambiguous pattern is treated as outlier. The current patterns in the data set form a specific distribution. The goal is to identify any new pattern that does not fall within the distribution of the current data set. Pattern recognition algorithms using a distance metric to make classification decisions can identify outliers by developing a distance threshold (sometimes called a measure of proximity). For example, in LVQ or NN, the Euclidean distance between the new pattern and each pattern in either the training set or the hidden layer is computed. If the closest distance is greater than some preset threshold, the new pattern is classified as an outlier.

Another approach to outlier rejection is to use the a posteriori probability [21]. If the probability of a pattern being a member of any class if less than 90% (or some probability that is application-dependent), it can be rejected as an outlier. This approach is compatible with both two-class and multi-class problems. The only disadvantage to using the probability to detect outliers can be illustrated by way of example. Figure 8 shows a simple two-dimensional data space with two data classes represented by Xs and Os for class 1 and 2, respectively. In this example, there are two outliers, A and B. Outlier B would be easily found as an outlier because the PNN (or any other of the algorithms) would assign it a 50% probability of being a member of either class. However, outlier A presents a more difficult situation. Because this outlier is so far away from class 1 in the data space, despite clearly being an outlier, it would be assigned a high probability of being a member of class 2 because the Bayes rule requires that the summation of class probabilities equal 1. Thus, in addition to the probability criterion for outlier detection, a second approach based on distance is needed.

For a PNN, the distance based outlier rejection approach was used by Bartel et al. in the analysis of nuclear powerplant transient diagnostics [43]. The challenge to outlier rejection in a PNN is the choice of the cutoff criterion. In PNN outlier rejection, the cutoff threshold operates on the aggregate score of the summation neurons (see Fig. 2). If this value is less than the cutoff threshold, the pattern is considered an outlier and the operator is alarmed.This task was greatly simplified in the classification of nuclear powerplant transient signals because the values stored in the summation neurons of the PNN were equal to 0.0 for outliers. Thus, they could easily set the threshold to 0.0 without further investigation. However, this is not a general solution to the problem. Most chemical sensor applications will require much tougher criteria for rejecting outliers and with current technology cannot be determined in advance. The goal of the following research was to develop a general protocol for setting the optimal rejection level for a PNN-based classifier (e.g., PNN, PNN5, LVQ-PNN, or LVQ-PNN5) of chemical sensor array data.

Monte Carlo Simulations

Monte Carlo simulations studies were used in this work to determine outlier rejection threshold levels. Monte Carlo methods comprise a branch of experimental mathematics that is concerned with experiments on random numbers [44]. In this work, random pattern vectors were generated and presented to the PNN. Many of these patterns (e.g., outliers A and B in Fig. 8) are not located near (in terms of Euclidean distance or dot product distance) any of the "true" data. Thus, they will not fall within the given distribution of the existing patterns. By generating thousands of random patterns, a cutoff level that will reject a majority of these patterns can be determined. If the rejection threshold is set too high, the probability of rejecting a pattern falling within the statistical bounds of the distribution increases. Selecting a rejection level too small decreases the chances that a potential outlier will be flagged as an outlier. Because we are interested primarily in developing protocols for the distance-based outlier detection, the probability-based outlier detection is not included in these studies. However, for actual implementation on sensor systems in the field, both methods are recommended.

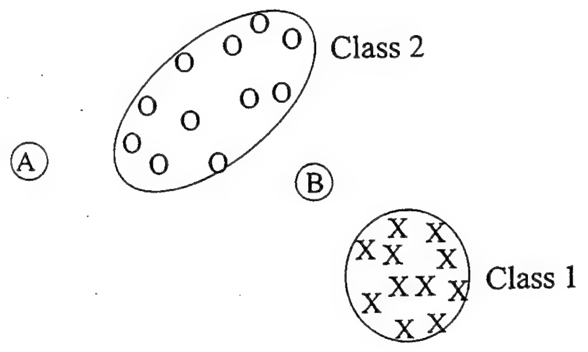


Fig. 8 — Conceptual picture of a two-dimensional pattern space consisting two data classes represented by Xs and Os, respectively, and two outlier patterns, A and B

Considerations for Setting the Rejection Threshold

The optimal rejection threshold will certainly be dependent on the data set and application. Each pattern space will have a certain distribution, thereby potentially requiring a different rejection criterion. In a PNN, the optimal rejection level will be based on the number of patterns (or neurons in the hidden layer) in each class and the kernel width. Any changes made to either the hidden layer or training set or the kernel width will require a new choice for the rejection threshold. The rejection threshold selection is also application-dependent. By changing the target rejection goal (discussed below), the rejection can be made either more lenient or more stringent, depending on the application. If the patterns for the sensor system are extremely stable over time and experiment conditions, a more stringent target rejection goal can be used because the patterns collected in the field are less likely to be different than those in the training data set. Thus, the definition of an outlier can be made more strict. Likewise, for some applications, the definition of an outlier can relaxed and a more lenient target rejection goal can be used.

Experimental

The SAW2 data set was used to study methods for determining the best rejection threshold. For this data set, additional data were available for chemical vapors that are not similar to any of the those included in the original training and prediction subsets and thus should be flagged as an outliers. One pattern vector corresponding to toluene and another for octane were used. These organic vapors are not similar to any of the organosulfur and organophosphorus vapors included in the SAW2 data set.

We empirically determined that a rejection threshold r of σ^4 was a good initial estimate. The optimized LVQ-PNN5 (run 3 in Table 12) was presented with 10,000 random normalized pattern vectors. The strengths of the summation neurons were tested. If the aggregate summation neuron strength was less than r, the pattern was labeled an outlier. The target rejection goal for this implementation was to find between 85% and 90% of the random patterns as outliers. If less than 85% of the patterns were rejected, then r is not strict enough and was increased by 20% and the simulation was repeated. If greater than 90% of the random pattern vectors were determined to be outliers, then r is too strict and was decreased by 20% and the simulation was repeated. The choice of 85 to 90% for the target rejection goal is somewhat arbitrary and definitely application-dependent. More research in this area needs to be done.

Results

By using the Monte Carlo simulation procedure, the rejection threshold for the SAW2 data set, using the optimized LVQ hidden layer for the PNN and a kernel width of 0.0465, was found to be 0.0000468 (4.68×10^{-6}). For this threshold, the LVQ-PNN5 approach rejected 89.04% of the random patterns in the simulation. The variation in the percentage of random patterns rejected was determined to be approximately +/- 0.5% based on replicate experiments. Of course, with a large number of random patterns ($n = 10000$), the variation would be expected to be quite small [44].

To test this threshold on real data, two experiments were performed. First, the entire SAW2 data set was presented to the same LVQ-PNN5 configuration to determine if any of the original patterns would be flagged as outliers. Two patterns in the original data set (2 out of 664 = 0.3%) were found to be outliers. These same two patterns (VX data class) can be seen as outliers from the PCA plot in Fig. 6. Because these patterns were present in the prediction subset, the LVQ training procedure could not account for them. Thus, labeling these two patterns as outliers is acceptable, although not ideal. The second experiment involved the two pattern vectors taken from SAW data for an exposure to toluene and octane. Because these patterns (i.e., their chemical signatures) are much different than the patterns in SAW2 data set, they should be rejected as outliers. After these patterns were presented to the same LVQ-PNN5 configuration as above, they were both flagged as outliers. Their aggregate summation neuron scores were both less than 1.3×10^{-8}.

Conclusions

The results from this part of the study indicate that the Monte Carlo-based procedure for determining the optimal rejection threshold is valid. Two experiments were performed to verify that only true outliers were rejected. A general procedure for selecting the rejection threshold is critical for applying a PNN-based pattern recognition scheme to chemical sensor array applications. By combining the distance-based and the probability-based outlier rejection schemes, the chances of a false alarm occurring will decrease.

Further work needs to done to better define what the appropriate choices are for the target rejection goal. Based on our knowledge of applying chemical sensor arrays to toxic vapor detection, a target of between 85 and 90% seems reasonable. However, these target values may not be optimal for other applications.

CONCLUSIONS

In the first part of this study, seven pattern recognition algorithms were compared based on their ability to meet the criteria for an ideal classifier. The PNN and LVQ approaches were both found to have good classification accuracy and met many of the qualitative criteria for an ideal classifier. The weakness of the LVQ approach is that it does not have a simple method for generating a confidence measure for each classification decision. The PNN algorithm has large memory and computational requirements, which could potentially make implementation on a microprocessor difficult.

In the second part of this report, a new pattern recognition scheme (LVQ-PNN5) based on the PNN and LVQ algorithms was presented. The classification accuracy of this approach was compared with seven other pattern recognition algorithms. The results demonstrated that LVQ-PNN5 is an excellent pattern recognition algorithm for chemical sensor array systems. In terms of the criteria for an ideal pattern recognition algorithm for chemical sensor array systems, the LVQ-PNN5 approach scores well on all points. The LVQ-PNN5 algorithm features several improvements over the basic PNN and LVQ methods and is better suited for implementation on a microcontroller for operation on sensor systems in the field.

A novel, general procedure for selecting the optimal rejection threshold for a PNN-based algorithm using Monte Carlo methods was also presented. An outlier detection scheme greatly reduces the probability of false alarms resulting from ambiguous or spurious sensor signals. This outlier detection approach was implemented for a LVQ-PNN5 classifier and found to consistently reject ambiguous patterns.

Overall, the LVQ-PNN5 algorithm gives the scientist or engineer the ability to develop a powerful classification algorithm with little or no user interaction. This approach is based on sound theory and experimental evidence. Developing a classification algorithm for a particular sensor array system can be a difficult and frustrating task. However, the research presented in this report illustrates that it does not have to be.

ACKNOWLEDGMENTS

This research was performed while Ronald E. Shaffer was a National Research Council-Naval Research Laboratory Postdoctoral Research Associate. Dr. Mark Anderson is thanked for helpful discussions of PNN and providing the two simulated data sets. Dr. David Venezky is thanked for his interest in this work.

REFERENCES

1. S. Singh, E.L. Hines, and J.W. Gardner, "Fuzzy Neural Computing of Coffee and Tainted-water Data From an Electronic Nose," *Sens. Actuat.* **30**, 185-190 (1996).

2. S.J. Haswell and A.D. Walmsley, "Chemometrics—The Key to Sensor Array Development," *Analyt. Proc.* **28,** 115-117 (1991).

3. H. Sundren, I. Lundstrom, F. Winquist, I. Lukkari, R. Carlsson, and S. Wold, "Evaluation of a Multiple Gas Mixture with a Simple MOSFET Gas Sensor Array and Pattern Recognition," *Sens. Actuat. B* **2**, 115-123 (1990).

4. J.R. Stetter, P.C. Jurs, and S.L. Rose-Pehrsson, "Detection of Hazardous Gases and Vapors: Pattern Recognition Analysis of Data from an Electrochemical Sensor Array," *Analyt. Chem.* **58**, 860-866 (1986).

5. J.W. Grate, S.L. Rose-Pehrsson, D.L. Venezky, M. Klusty, and H. Wohltjen, "Smart Sensor System for Trace Organophosphorus and Organosulfur Vapor Detection Employing a Temperature-Controlled Array of Surface Acoustic Wave Sensors, Automated Sample Preconcentration, and Pattern Recognition," *Analyt. Chem* **65**, 1868-1881 (1993).

6. E.T. Zellers, S.A. Batterman, M. Han, and S.J. Patrash, "Optimal Coating Selection for the Analysis of Organic Vapor Mixtures with Polymer-Coated Surface Acoustic Wave Sensor Arrays," *Analyt. Chem.* **67**, 1092-1106 (1995).

7. J.W. Grate and M.H. Abraham, "Solubility Interactions and the Design of Chemically Selective Sorbent Coatings for Chemical Sensors and Arrays," *Sens. Actua. B* **3**, 85-111 (1991).

8. R.A. McGill, M.H. Abraham, and J.W. Grate, "Choosing Polymer Coatings for Chemical Sensors" *Chemtech* **24**, 27-37 (1994).

9. W.P. Carey, K.R. Beebe, B.R Kowalski, D.L. Illman, and T. Hirschfeld, "Selection of Adsorbates for Chemical Sensor Arrays by Pattern Recognition," *Analyt. Chem.* **58**, 860-866 (1986).

10. T. Aishima, "Discrimination of Liquor Aromas by Pattern Recognition Analysis of Responses from a Gas Sensor Array," *Analytica Chimica Acta* **243**, 293-300 (1991).

11. R.E. Shaffer and G.W. Small, "Improved Response Function for the Simplex Optimization of Piecewise Linear Discriminants," *Chemometrics Intell. Lab. Syst.* **32**, 95-109 (1996).

12. J.R. Stetter, M.W. Findlay, K.M. Schroeder, C. Yue, and W.R. Penrose, "Quality Classification of Grain Using a Sensor Array and Pattern Recognition," *Analytica Chimica Acta* **284**, 1-11 (1993).

13. C. Di Natale, F. Davide, A. D'Amico, "Pattern Recognition in Gas Sensing: Well-stated Techniques and Advances," *Sens. Actua. B* **23**, 111-118 (1995).

14. J. Auge, P. Hauptmann, J. Hartmann, S. Rosler and R. Lucklum, "Versatile Microcontroller Gas Sensor Array System Using the Quartz Crystal Microbalance Principle and Pattern Recognition Methods," *Sens. Actua. B* **26-27**, 181-186 (1995).

15. M.R. Anderson and D.L. Venezky, "Investigation of Probabilistic Neural Networks for Sensor Array Pattern Recognition, " NRL MR/6170--96-7798 (1996).

16. S.L. Rose-Pehrsson, J.W. Grate, D. DiLella, and M. Klusty, "A Smart Sensor System Utilizing a Surface Acoustic Wave Vapor Sensor Array and Pattern Recognition for Selective Trace Organic Detection Part II," NRL Memo. Rept. 6917, Nov. 1991.

17. J.W. Grate, S.L. Rose-Pehrsson, D.L. Venezky, M. Klusty, and H. Wohtjen, "A Smart Sensor System Utilizing a Surface Acoustic Wave Vapor Sensor Array and Pattern Recognition for Selective Trace Organic Detection Part 1," NRL Memo. Rept. 6916, Nov. 1991.

18. R.A. McGill, C. Chon, and D.L. Venezky, "Design & Evaluation of SAWCAD Performance Part 1: Field Tests in South Korea 3rd-7th November 1994," NRL Memo. Rept. in preparation.

19. R.A. McGill et al., "Design & Evaluation of SAWCAD Performance Part 2: Chemical Agent Test & Evaluation, " NRL Memo. Rept., in preparation.

20. R.E. Shaffer, S.L. Rose-Pehrsson, and R.A. McGill, "Automated Identification of Chemical Warfare Agents using Surface Acoustic Wave Chemical Sensors," Proceedings of the 1996 ERDEC Scientific Conference on Chemical/Biological Defense, pp. 939-945.

21. M.P. Derde and D.L. Massart, "Supervised Pattern Recognition: The Ideal Method?," *Analytica Chemica Acta* **191**, 1-16 (1986).

22. D. Michie, D.J. Spiegelhalter, and C.C. Taylor, *Machine Learning, Neural, and Statistical Classification* (Ellis Horwood, New York, 1994).

23. C. Klawun and C.L. Wilkins, "Optimization of Functional Group Prediction from Infrared Spectra Uusing Neural Networks," *J. Chem. Infor. Comp. Sci.* **36**, 69-81 (1996).

24. J.T. Tou and R.C. Gonzalez, *Pattern Recognition Principle*s (Addison-Wesly, Reading, MA, 1974).

25. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973).

26. C.M. Bishop, *Neural Networks for Pattern Recognition* (Oxford University Press, New York, 1995).

27. D.L. Massart, B.G.M. Vandeginste, S.N. Deming, Y. Michotte, and L. Kaufman, *Chemometrics: A Text Book* (Elsevier, Amsterdam, 1988).

28. O. Strouf, *Chemical Pattern Recognition* (Research Studies Press, Letchworth, UK, 1986).

29. W.J. Krzanowski, *Principles of Multivariate Analysis* (Oxford University Press, New York, 1988), p. 359.

30. T. Masters, *Practical Neural Network Recipes in C++* (Academic Press Inc., Boston, MA, 1993).

31. T. Masters, *Advanced Algorithms for Neural Networks* (John Wiley, New York, 1995).

32. D.F. Specht, "Probabilistic Neural Networks," *Neur. Net.* **3**, 109-118 (1990).

33. D.J. Hand, *Kernel Discriminant Analysis* (Research Studies Press, Letchworth, UK, 1982).

34. T. Kohonen, *Self-Organization and Associative Memory* (Springer-Verlag, Berlin, 1988).

35. B. Kamgar-Parsi and B. Kamgar-Parsi, "Integration of Detection and Classification of Signals from Neural Networks, NRL NCARI Rept. AIC-93-028.

36. A.S. Bangalore, G.W. Small, R.J. Combs, R.B. Knapp, R.T. Kroutil, C.A. Traynor, and J.D. Ko, "Automated Detection of Trichloroethylene by Fourier Transform Infrared Remote Sensing Measurements," *Analyt. Chem.* **69**, 118-129 (1997).

37. W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, *Numerical Recipes in C: the Art of Scientific Computing* (Cambridge University Press, Cambridge, UK, 1992).

38. S.R. Johnson, J.M. Sutter, H.L. Engelhardt, P.C. Jurs, J. White, J.S. Kauer, T.A. Dickinson, and D.R. Walt, "Identification of Multiple Analytes Using an Optical Sensor Array and Pattern Recognition Neural Networks," *Analyt. Chem.* **69**, 4641-4648 (1997).

39. P. Burrascano, "Learning Vector Quantization for the Probabilistic Neural Network," *IEEE Trans. Neur. Net.* **2**, 458-461 (1991).

40. Y. Chtioui, D. Bertrand, and D. Barba, "Reduction of the Size of the Learning Data in a Probabilistic Neural Network by Hierachical Clustering Application to the Discrimination of Seeds by Artificial Vision," *Chemometics Intellig. Lab. Syst.* **35**, 175-186 (1996).

41. M.A. Kraaijveld, "A Parzen Classifier with an Improved Robustness Against Deviations Between Training and Test Data," *Pat. Recog. Lett.* **17**, 679-689 (1996).

42. H. Demuth and M. Beale, *Neural Network Toolbox User's Guide* (Mathworks Inc., Natick, MA, 1995.

43. Y. Bartal, J. Lin, and R.E. Uhrig "Nuclear Power Plant Transient Diagnostics Using Artificial Neural Networks That Allow 'Don't-Know' Classifications," *Nuclear Tech.* **110**, 436-449 (1995).

44. O.A. Guell and J.A. Holcombe, "Analytical Applications of Monte Carlo Techniques," *Analy. Chem.* **62**, 529A-542A (1990).

ACRONYMS

ALLOC	similar to PNN
ANN	artificial neural network
BDA	Bayes Linear Discriminant Analysis
BP	back propagation
BP-ANN	back propagation artificial neural networks
CEE	chloro ethyl ether
CV	cross validation
CV-OPT	cross validation with univariate optimization
CW	chemical warfare
DCP	dichloropentane
DMMP	dimethyl methyl phosphonate
GA	ethyl N,N - dimethylphosphoramidocyanidate
GB	isopropyl methylphosphonofluoridate
GD	pinacolyl methyl phosphofluoridate
HD	bis(2-chloroethyl) sulfide
LDA	linear discriminant analysis
LVQ	learning vector quantization
NN	nearest neighbor
PCA	principal components analysis
PDF	probability density function
PNN	probabilistic neural network
RBF	radial basis function
RN	reciprocal neighbors
SAW	surface acoustic wave
SIMCA	soft independent modeling of class analogy
VX	O-ethyl-S-(2 isopropylaminoethyl) methyl phosphonothiolate
WTA	winner-take-all